Towards an Ontological Framework for Execution Monitoring

Riley MOHER [a], Michael GRÜNINGER [a],

[a] *Department of Mechanical and Industrial Engineering, University of Toronto, Ontario, Canada M5S 3G8*

Abstract. Business Process Management (BPM) is a useful management framework that is being adopted by a growing number of enterprises. However, the current state of BPM lacks any true process semantics. Though there has been work focusing on incorporating semantics into BPM, they lack the expressiveness to support reasoning capabilities for execution monitoring. Through a case study in compliance issue remediation at a major Canadian bank, we identify the concept of status values as a key and novel component of execution monitoring in a semantic BPM environment. Existing process ontologies cannot accurately represent status values due to the limitations of semantic web ontologies and the use of event ontologies over process ontologies. To address these gaps, we developed the compliance issue remediation ontology (CIRO) based on the Process Specification Language (ISO 18629). The underlying framework of CIRO represents status values as occurrence facts. In this representation, workflow is agnostic of status, which facilitates a simplified approach to representing business processes at various levels of granularity. Furthermore, this framework has broader implications for process ontology research including process decomposition and abstraction. Extensions for this work include process mining and linguistic extensions to tense and aspect. Altogether, a fact-driven ontological understanding of status values is not only a significant step towards semantic execution monitoring, but a positive development towards how we should think about process.

Keywords. status value, execution monitoring, semantic business process management, ontology, process specification language

Introduction

Business Process Management (BPM) is a useful tool for enterprises to understand their processes, and is being adopted in many industries. While BPM enables businesses to better understand their process inventory, this approach lacks the kind of semantics required for reasoning about processes. The development of novel semantic BPM techniques and methods is an important area of research to enable consistent enterprise management of process documentation, repositories, and systems. An effective semantic BPM system should be one that supports existing enterprise tools in an unobtrusive way.

Execution monitoring is one of the tools enterprises have in their BPM toolkit for controlling risk, ensuring compliance, and measuring process performance. While existing ontological approaches for execution monitoring exist, their narrow scope and reliance on semantic web technologies limits their reasoning capabilities. Current approaches abstract the fundamental parts of process knowledge, or conflate different notions of the execution of a business process, like states and events. This paper introduces a new view of execution monitoring, in which the central research question is: "What is a business status value and how should it be represented?". The ubiquity of status values, across various granularities and information systems validates this question as a meaningful step in the creation of practicable semantic BPM. In order to answer this question, we take an ontological approach based on the Process Specification Language (PSL) [1].

We also provide motivation for the representation of business status values through a case study at a major Canadian bank and create representational requirements from this case study. Related work will be presented to set the stage for our solution and to describe how these approaches fail to meet the representational requirements of status values. Subsequently, our approach for representing status value as occurrence facts will be detailed and evaluated in a real business context through the Compliance Issue Remediation Ontology (CIRO). We will then explain the broader implications of our representation in terms of general process semantics. Subsequently, possible extensions to this work through process mining and linguistics will be explored.

1. Motivation: Compliance Issue Remediation

Execution monitoring is an important tool to manage and improve business processes as they are executed. The motivation behind this paper comes from an execution monitoring case study at a major Canadian bank, hereby referred to as The Bank. The case study in this section will inform the representational requirements for the concept of status values.

At The Bank, procedures and policies exist to ensure the Bank complies with any regulatory requirements, and that any regulatory breaches or risks are identified and resolved. Compliance issue remediation responsibilities fall to employees across the enterprise, from the branch to the executive level. Identified compliance issues may range in complexity, and their remediation may be complex and involve multiple teams of individuals. Action plans are put in place to describe the steps to be completed for remediation of an issue, with some issues requiring multiple action plans. Both compliance issues and individual action plans are monitored.

The compliance issue remediation process is managed within a software system accessed by employees at varying levels, from employees remediating individual issues to those overseeing the remediation of issues across entire departments. To help monitor the remediation process, high-level status values are put in place to classify the low-level remediation process into a handful of progress markers. These status values apply both to compliance issues and to action plans. For example, when a compliance issue has been automatically submitted for review after a change has been made to the issue, but before the review has been completed, it will have a status value of "Revise - Under Review".

Status values serve as a simple representation of process workflow history, ideal for executives in The Bank. It is important to note, however, that status values are defined a priori, as part of process specification, as opposed to having a purely event-driven definition. Status values, however, are not necessarily defined from a singular business perspective. During compliance issue remediation, there may exist multiple sets of status values corresponding with different business perspectives and different levels of process granularity. These status values may not even be formally defined, for example, as a column in a relational database, but used as part of the business vocabulary. The ubiquity of status values in execution monitoring then, is a significant motivator for a means of formally representing status values in terms of the underlying processes they apply to.

2. Status Value Representation Requirements

Now equipped with an understanding of the compliance issue remediation process at the Bank, we can lay out the key requirements for representing status values. The key requirements for representing status values can be defined in three aspects: relationship to workflow, object-orientation, and the role of process atomicity.

2.1. Relationship to Workflow

The fundamental requirement of the representation of status values is the connection to process workflow, as equivalent to an arbitrary grouping of execution steps in a process workflow. In order to properly fulfil this requirement, accurate representation of status values requires that workflow be agnostic of status value. Since the definition of status values can be done at different levels of process granularity, maintaining the autonomy of process workflow definitions and status values definitions ensures that different business perspectives do not burden the management of process workflow definitions. If the steps of a business process were to be conditioned on systems-level status values, the definition of executive-level status values would be inaccessible without an understanding of the lower-level status values.

Status values, while being defined in terms of workflow, are not themselves higher-level workflows. In process mapping, parent and children activities enable the abstraction of many task-level activities into a high-level process with just a few steps. The relationship of status values and workflow is more flexible than that of parent and children activities. Status values are more flexible in that the set of all status values may not cover the entire continuity of process flow, and multiple status values may hold simultaneously. In order to maintain this relationship, status values must not be defined in terms of other status values, and must not be dependent on other status values. Status values are aggregations of process flow, rather than abstractions.

2.2. Object-Orientation

Status values are described in the context of the entities flowing through business processes, and just like the relationship between workflow and status values, this relationship has nuances that must be accurately captured. This relationship is born from the fact that business processes begin with some input and, throughout the process, that input is transformed into some final output(s). In an execution monitoring environment, there may be multiple entities that flow through a process, or a single central entity. To illustrate the important aspects of this relationship, we will describe the distinction between the more traditional concept of states and status values as they apply to objects.

States describe properties of an object which are observable outside of the process the object is involved in, while status describes properties of an object *as it is involved in a process*. For example, after a part has been painted, the part is in the painted *state*, as this property may be observed outside of the manufacturing process. On the other hand, a document has the status value 'under review' after the review process is initiated, but before it has finished, the status of the document cannot be assessed without information of the review occurrence.

2.3. Role of Process Atomicity

The question of atomicity in process ontology applications (that is, deciding which activities are the indivisible units of a process) is a decision which should not affect the way status values are represented. Considering the lack of prominent research into the role atomicity plays in process ontologies, introducing any kind of dependence between status values and atomicity may create unnecessary additional complications which are not the focus of this paper.

In order to maintain a consistent semantics for status values, this flexibility in the granularity of status value definitions requires constraints on how the supporting process ontology supports aggregation. For example, to allow for status values to be defined in terms of how someone grabs objects or in terms of how a person moves their fingers to grab objects, the process ontology must specify the set of finger movement actions that constitute grabbing actions. Therefore the range in granularity of status values is necessarily linked to that of the process ontology.

2.4. Competency Questions

In addition to the above representational requirements, we can create competency questions [2] to evaluate any ontological representations of status values.

- What is the status value(s) of the object at a time point?
- What occurrences and/or partial occurrences constitute the object having the status value?
- What are all the objects having a certain status value at a point in time?
- What are the next possible status values of an object?

3. Related Work

Having identified the requirements to represent business status values, understanding the current state of semantic business process management and process ontologies will set the stage for our framework. The representation of status values has not been explicitly identified as a key component in these works, so our comparison will not be a direct comparison of approaches. Instead, we will detail why existing frameworks do not permit the representation of status values as we have specified.

3.1. SBPM

Hepp et al. [3] introduce Semantic Business Process Management (SBPM) as an approch to manage the execution of IT-supported busines operations from a business experts process view. The fundamental motivation behind SBPM is one that is very relevant to our work - the IT/Process divide. The IT process divide is the idea that there are two main process spheres, the business expert perspective and the IT implementation perspective, and BPM alone cannot facilitate translations between the spheres. SBPM, however, relies on Semantic Web Services (SWS) for knowledge representation, which does not provide the level of expressiveness suitable to fulfill the representational requirements of status values.

3.2. SUPER

One of the most significant pieces of research in the SBPM field is the Semantics Utilized for Process management within and between Enterprises (SUPER) [1] Project, which seeks to raise BPM from the IT level to the business level through the application of semantic web services. Contrary to our work, SUPER's scope is far-reaching and transformative, covering the full BPM environment of an enterprise. The SUPER stack [4], the primary ontological portion of SUPER, is designed to represent the core business process aspects of SBPM. The full SUPER ontology stack can be seen in Figure 1.

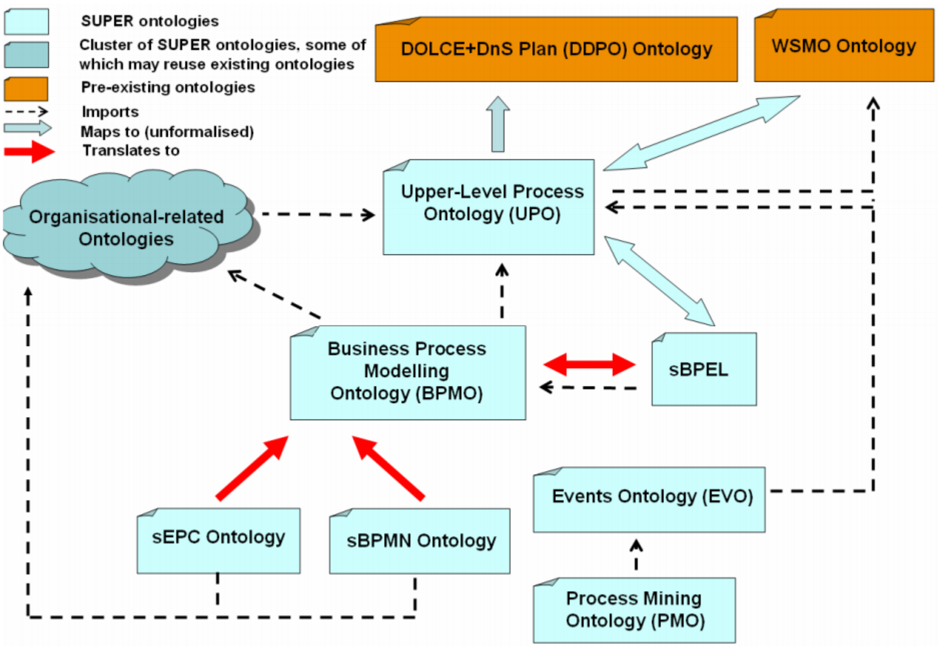

Figure 1. Process Modelling Ontology Stack in SUPER (taken from [4])

The SUPER stack consists of ontologies designed specifically to integrate with process modeling notations and IT systems, including BPMN[2], EPC [5] and event log systems. Similarly to SBPM, SUPER's complex architecture is a result of its limitations in terms of its expressiveness. The Events Ontology (EVO) within the SUPER stack, is most relevant to our work.

---

[1] `https://web.archive.org/web/20091213192728/http://www.ip-super.org/component/option,com_frontpage/Itemid,1/`

[2] `http://www.bpmn.org/`

The SUPER stack contains the Business Process Modelling Ontology (BPMO), functioning as an enterprise-specific extension of the Upper Process Ontology (UPO). Process modelling notations and systems, including BPMN and EPC are each related to the BPMO through translations. However, EVO is placed in a unique position within the stack, as extending the UPO rather than having a translation between BPMO and EVO. Pedrinaci et al [6] present EVO as a component for an ontological approach to process monitoring and mining. Much like the name suggests, EVO is built around events as the foundational unit, with causality relations between events, and distinct event classes, in-line with event log systems. EVO's placement within the stack makes sense for an SBPM-based approach to execution monitoring, as the limited expressiveness of SWS necessitates events to be defined as an extension of an upper process ontology. Within this framework, however, the representational requirements of status values will never be met, as event semantics and process semantics are distinct. Because of SUPER's reliance on SWS, the expressiveness of the main process ontology does not allow robust inference that would enable the translation between process workflow and an event log perspective. Status value representation requires an ontological framework that does not separate events from a process ontology, but a semantically powerful process ontology as the base of the framework.

3.3. Process Ontologies: Fluents and States

There is a significant distinction to be made between the existing notions of states/fluents and status values, especially considering the reliance of various process ontologies on these states/fluents. Status values are not a replacement for states/fluents, but an additional perspective on how process knowledge may be represented for the purposes of monitoring.

The intuitive response to representing status values may be to classify them as fluents, as described in the Process Specification Language (PSL) [1], or Situation Calculus [7]. Fluents are states of the world which change according to the occurrence of activities, and constraints may centre around these fluents. Trying to fit status values into fluent definitions would be inconsistent with the representational requirements previously laid out. Fluents describe states of the world which enable processes to progress, whereas status values describe the progression of processes. Status values do not enable new situations to take place; to define status values using fluents would be to create frivolous fluents.

In the work of Sanfilippo [8], an ontological analysis of BPMN events and activities in the context of the DOLCE [9] taxonomy of perdurants is carried out. Sanfilippo's analysis identifies an implicit state behind catch events, one that is homeomeric and cumulative, consistent with DOLCE's notion of states. These hidden states are more closely related to fluents than to status values. The hidden state 'ready to receive' either has a specific trigger, or is exogenous to the process, and is assumed to be the default state, and the receiving activity is predicated on this state. It is important to note that this distinction, however, is not clear-cut within DOLCE's taxonomy - status values and hidden states may both be classified under the DOLCE concept of state.

4. The Compliance Issue Remediation Ontology

The CIRO project began as an effort to align event logs with database representations of compliance issues through the remediation process. At the beginning of the project, the focus was more towards the application, specifically to use process mining techniques to check the alignment of event logs with categorical variables, using methods based on [10]. However, upon creating competency questions for the ontology to support this process mining, the problem of representing status values was recognized as the primary method for tracking issues for the issue remediation team at The Bank.

Understanding the concepts which were to be represented by the Ontology and understanding the current state of the compliance issue remediation process was supported by two sources. First, governance documents outlined compliance procedures and provided a detailed description of the compliance issue remediation process and associated terminology. Second, exports from the compliance oversight engine contained the logs from compliance issues and other data classifying these issues. From these sources, a conceptual model of the compliance issue remediation process was created and served as the reference for the formalization of the issue remediation process.

In the following subsections, we will describe the ontological foundations, structure, and axiomatization of CIRO.

4.1. Ontological Foundation

For the axiomatization of CIRO, first-order logic and the Process Specification Language (PSL) were used, with reference especially to [11]. PSL is an ontology designed to facilitate correct and complete exchange of process information among manufacturing systems [1]. The decision to use PSL was motivated primarily by its expressiveness, something that is necessary to accurately represent status values. The modules of PSL that were utilized are PSL-Core[1], Subactivties[2], Complex Activities[3], and Subactivity Occurrences[4].

PSL-Core is the minimal module within PSL, which axiomatizes the primitive concepts of PSL: activity, activity occurrence, object, and timepoint. These four concepts form a solid base for reasoning about business processes. Activities are the units that typify the steps of a process, and they may occur many times or not at all. Activity occurrences and objects are related to timepoints through a beginning and an end point of the object/occurrence. The objects within PSL-Core may participate in activity occurrences at some specific timepoint. Unlike other approaches that may contain a complex taxonomy of process units, PSL's semantically-robust base allows for complex definitions of other process constructs, like status values, to be driven by inference. This provides a more transparent view of a process, and hides complexity behind inference, rather than relying on application-specific extensions of the process ontology.

The subactivity module within PSL allows for a very important and desirable feature of process modelling: composition of activities. PSL does so in a very flexible way which allows for discrete partial orderings of activities. Having this flexibility is crucial in the

[1] `colore.oor.net/psl_core/psl_core.clif`
[2] `colore.oor.net/psl_subactivity/subactivity.clif`
[3] `colore.oor.net/psl_complex/complex.clif`
[4] `colore.oor.net/psl_actocc/actocc.clif`

execution monitoring environment where the action plans to address compliance issues will not always have a strict ordering.

The complex activity and subactivity occurrence modules are a crucial part of CIRO because of the object-oriented nature of the remediation process. The *min_precedes*($s1, s2, a$) ordering relation within the complex activity module defines a linear ordering of subactivities, and the subactivity occurrences module axiomatizes the corresponding occurrents, which will be crucial for specifying issue-specific orderings within CIRO.

4.2. Structure

CIRO consists of three main parts: structural definitions, occurrence constraints, and status value definitions.

Structural definitions provide semantics for the core process elements of CIRO, by utilizing the ontological base. These definitions are an extension of enterprise knowledge and extend the fundamental business concepts of compliance issue remediation. For the purposes of the CIRO project, this element focuses on just a few concepts and does not have a full enterprise ontology behind it. However, the integration of this framework into a more knowledge-modelling mature organization would utilize many concepts from existing enterprise ontologies and build out their process semantics through PSL.

Occurrence constraints describe the ordering of occurrences in the compliance issue remediation process. This ordering applies on a per monitoring item basis, consistent with the object orientation requirement of status values. This component, on its own, covers the same information that a process modelling language would, but has a much deeper semantics within the broader ontological framework. There are many different kinds of constraints that determine how a process unfolds, and PSL provides a thorough semantics for various kinds of constraints. For a thorough explanation of these constraint mechanisms within PSL, see [11].

Finally, status value definitions define status values for individual monitoring items. Status values are driven by inference, and the status value component of CIRO is designed to be modular. Status values are essentially equivalent to an arbitrary point on the activity tree of a monitoring item. Status value axioms maintain autonomy and do not constrain any other axioms within the ontology, nor do any other axioms depend on status value definitions.

The signature of CIRO is the enumeration of all the relations it uses, in addition to those included in the PSL ontological base. Table 1 contains the CIRO signature.

It is important to note the ontological commitments that are evident in CIRO's signature. First, the business concepts which are the main classes of entities within the ontology are unary relations that specify types, with the exception of the $monitor\_activity$ relation. The monitor activity relies on PSL's complex activities to define the set of all activities which apply to a monitoring object. The *assigned_to* relation is the only binary relation within CIRO, and it defines the relationship between issues and action plans, it is part of the structural definitions and does not rely on PSL, but enterprise knowledge instead. The monitoring activities are the fundamental steps of the monitoring process and, consistent with PSL, the unary relation's argument specifies the monitoring object on which they operate. The binary $status\_of$ relation is built on top of PSL as more of a terminological extension. The specific uses of CIRO's relations will be demonstrated in the Axiomatzation section.

Table 1. CIRO Signature

Category	Relation	Definition
Business Concepts	$compliance\_issue(i)$	i is a compliance issue
	$action\_plan(a)$	a is an action plan
	$monitoring\_object(x)$	x is a monitoring object
	$monitor\_activity(a,x)$	a is the complex activity consisting of all the monitoring activities of the monitoring object x
	$step\_for\_remediation(s)$	s is a step for remediation
Concept Relations	$assigned\_to(a,i)$	a is the action plan that has been assigned to compliance issue i
Monitoring Activities	$material\_change(x)$	a material change activity performed on x
	$effective\_challenge(x)$	an effective challenge activity performed on x
	$validation(x)$	a validation activity performed on x
	$approval(x)$	an approval activity performed on x
	$rejection(x)$	a rejection activity performed on x
	$completion(x)$	a completion activity performed on x
Status Value	$status\_of(x,t)$	the status value of x at timepoint t

4.3. Axiomatization

In this section, we will present the modules of CIRO and the critical axioms from them. The axioms presented for each section are a subset of the full axioms[1], and they are only those which are sufficient for demonstrating the broader framework behind CIRO. Axiomatization is done in first-order logic using Common Logic[2]. Included in this section is CIRO-core, CIRO-XM, and CIRO-status. CIRO-core[3] is the set of axioms covering the structural definitions and ordering constraints for compliance issue remediation. CIRO-XM[4] contains the axioms which provide the basic execution monitoring semantics for compliance issue remediation, but their general form may be extended to all execution monitoring problems. CIRO-XM shares the fundamental concepts of compliance issues and action plans with CIRO-core, but does not import its axioms. In a more knowledge-modelling mature organization, both CIRO-core and CIRO-XM would import axioms from an enterprise ontology that includes non-process semantics of compliance issues. CIRO-status[5] imports both CIRO-core, which itself imports CIRO-XM, and contains the semantics for all the possible status values in our compliance issue remediation case study.

4.3.1. CIRO-core

CIRO-core consists of the axioms which are specific to the compliance issue remediation process, including the structural definitions which provide process semantics for business

[1] `colore.oor.net/ciro/ciro.clif`
[2] ISO 24707
[3] `colore.oor.net/ciro/ciro-core.clif`
[4] `colore.oor.net/ciro/ciro-xm.clif`
[5] `colore.oor.net/ciro/ciro-status.clif`

concepts, and the ordering constraints which provide semantics for the process workflow of the compliance issue remediation process.

Compliance issues are objects

$$(\forall x) compliance\_issue(x) \supset object(x) \tag{1}$$

Action plans are assigned to exactly one compliance issue

$$\begin{aligned}(\forall x) action\_plan(x) \supset (\exists y) compliance\_issue(y) \wedge assigned\_to(x,y) \\ \wedge((\forall z) assigned\_to(x,z) \supset (z = y))\end{aligned} \tag{2}$$

Axioms (1) and (2) are some of the axioms which provide the process semantics for the fundamental concepts of CIRO. In this case they cover the business concepts related to the compliance issue remediation process. Generalizing the form of these axioms, other business concept axioms for a different application may provide process semantics for whichever business concepts are crucial to that specific execution monitoring environment.

Material change occurrences must be followed by an effective challenge occurrence

$$\begin{aligned}(\forall o_1, x, a)(occurrence\_of(o_1, material\_change(x)) \wedge monitor\_activity(a,x)) \\ \wedge legal(o_1) \supset (\exists o_2) \wedge occurrence\_of(o_2, effective\_challenge(x)) \\ \wedge legal(o_2) \wedge next\_subocc(o_1, o_2, a)\end{aligned} \tag{3}$$

Axiom (3) is one of the ordering constraint axioms of CIRO. These ordering constraints are determined by the understanding of the workflow of the underlying processes being monitored. Process documentation and/or models are useful materials in creating these axioms.

4.3.2. CIRO-XM

The CIRO Execution Monitoring (XM) module contains the axioms which are applicable to execution monitoring problems in general. The central concept in CIRO-XM is the monitoring object. Within CIRO, the objects that are monitored are compliance issues and actions plans, but the items which may be considered the monitoring objects will be specific to each execution monitoring application. If one wishes to extend our framework to their own particular execution monitoring problem, CIRO-XM is the primary module to be included.

Compliance issues and action plans are the monitoring objects of CIRO

$$\begin{aligned}(\forall x) monitoring\_object(x) \equiv \\ compliance\_issue(x) \vee action\_plan(x)\end{aligned} \tag{4}$$

Monitoring objects are assigned a unique monitor activity

$$\begin{aligned}(\forall x) monitoring\_object(x) \supset \\ (\exists a) monitor\_activity(a,x) \wedge ((\forall a_1) monitor\_activity(a_1,x) \supset (a_1 = a))\end{aligned} \tag{5}$$

Monitor activities are activities which apply only to monitoring objects

$$(\forall a, x) monitor\_activity(a,x) \supset activity(a) \wedge monitoring\_object(x) \tag{6}$$

Axioms (5) and (6) are general to any execution monitoring problem, whilst the definition in (4) reflects our application's structural definition axioms, with compliance issues and action plans being the monitoring objects. In another application, whichever business concepts are the monitoring objects will need to be made explicit in the form of (4).

4.3.3. CIRO-Status

Finally, CIRO-Status provides the process semantics of status values for the specific set of business processes in our case study. Status value axioms are structured in the form of conservative definitions: status values are logically equivalent to some set of occurrence fact(s). Two of CIRO's status value axioms are:

A monitoring object has status *Draft* at a point in time if and only if there has not yet been any effective challenge occurrences for that object

$$\begin{gathered}(\forall x,t)(status\_of(x,t) = \text{Draft}) \equiv \\ (\exists o,a) monitor\_activity(a,x) \wedge occurrence\_of(o,a) \\ \wedge (\forall o_1) occurrence\_of(o_1, effective\_challenge(x)) \\ \wedge subactivity\_occurrence(o_1,o) \wedge earlier(t, beginof(o_1))\end{gathered} \tag{7}$$

A monitoring object has status *Revise - Under Review* at a point in time if and only if there has been an occurrence of material change followed by an ongoing occurrence of effective challenge for that object

$$\begin{gathered}(\forall x,t)(status\_of(x,t) = \text{Revise - Under Review}) \equiv \\ (\exists o,a,o_1,o_2) monitor\_activity(a,x) \wedge occurrence\_of(o,a) \\ \wedge occurrence\_of(o_1, material\_change(x)) \wedge subactivity\_occurrence(o_1,o) \\ \wedge occurrence\_of(o_2, effective\_challenge(x)) \wedge subactivity\_occurrence(o_2,o) \\ \wedge next\_subocc(o_1,o_2,a) \wedge earlier(endof(o_1),t) \wedge earlier(t, endof(o_2))\end{gathered} \tag{8}$$

Status value axioms are defined through the binary relation status_of, which takes two arguments, the monitoring object x and the timepoint t. Status values are defined through the position of t relative to the occurrence facts of x's monitor activity. t may be related to these occurrence facts through an upper- or lower-bound, or an interval. (7) is an upper-bounded status value, the status value holds at every point before the subactivity occurrence of effective challenge. (8) is bounded from both sides, the status value holds between the subactivity occurrences of material change and effective challenge. The boundary occurrences will always be subactivity occurrences of x's monitor activity occurrence.

4.4. Evaluation

In this section we will revisit the representational requirements of status value to evaluate our status value representation. We will also elaborate on the significance of meeting each of these requirements.

Relationship to Workflow - To maintain autonomy from workflow, the $status\_of$ relation is not referenced within any occurrence constraint axioms or structural definitions axioms. The flexibility of status value definitions with respect to workflow is achieved through the bounding relationship of our status value representations. These different boundary permutations allow for status value multiplicity as well as discontinuities. State-based representations of status values treat status values like effects that are falsified or achieved after specific occurrences, violating workflow-status agnosticism and offering less flexibility. This is the most significant accomplishment of our framework, and the critical differentiator of our approach as compared with other ontological execution monitoring frameworks.

Object orientation - Object orientation is achieved through the monitor activity concept, in its axioms (5) (6) and its essential role in status value axioms. By employing the complex activity modules of PSL, we define status values in terms of a complex activity, which we define as the monitor activity, composed of the activities specific to each unique monitoring object. They key aspect with our kind of object orientation is that process semantics are necessarily connected to the monitoring objects. Rather than updating some object's label as the process progresses, or having semantics of a process abstraction, our approach keeps the underlying process semantics intact.

Role of process atomicity - In our status value definition axioms, the occurrents that define status values are not necessarily the atomic steps in a process. This is particularly significant for different levels of business process granularity.

5. Implications and Extensions

5.1. Process Abstraction vs Process Aggregation

Our representation of status value speaks to a greater phenomenon in process knowledge representation: abstraction vs aggregation. In many description logic based approaches, like SBPM [3] or SUPER [4], limited expressiveness warrants the creation of ontologies which have a laser-focus on one application. New highly specialized classes are defined around these applications, and they become abstractions of some lower-level process ontology, leaving the important link to those fundamental semantic building blocks behind. This abstraction approach is consistent with the predominant use of semantic web technology where robust semantics are second in priority to software integration and functionality. Our work hopes to shift the direction of semantic BPM towards the fundamental semantics of process instead of a disparate semantics of process and software systems.

Looking specifically at the representational requirements, having alternate answers to these questions and thinking about how our initial findings may be refined is an exciting avenue for future research. Especially in an enterprise environment, being able to represent process knowledge with these characteristics introduces a new level of transparency and could lead to the development of entirely new systems.

5.2. Process Mining

One of the immediately relevant extensions to our work is in process mining. With the beginnings of the CIRO project being rooted with this specific application in mind, it's a clear connection. Current work in process mining relies on frameworks with simple semantics, similar to existing semantic BPM approaches. Much like the way in which execution monitoring is approached, ontologies of limited semantics and narrow scope end up being applied for specific integration.

Especially prevalent in existing ontological process mining techniques is a reliance on events as a distinct class [6] [12] with no semantic basis in a foundational process ontology. Approaching process mining with a clear definition of status value and PSL as an ontological foundation would be a novel approach to ontology-based process mining.

5.3. Linguistic Applications: Status, Tense, and Aspect

One of the more abstract extensions to our work would be the relationship to the idea of tense and aspect. Although not immediately obvious, there are parallels that could be drawn between the understanding of process concepts, like fluents and status values, within a business context and within a general linguistic context. Analyzing simple cases like the present progressive ”John is walking” as compared to the present perfect ”John has walked” through a process ontology lens would be a novel approach.

Existing work which compares different linguistics understandings of tense [13] [14] employ ”tense logic”. Applying a process ontology view, equipped with the knowledge of status values as a distinct view of process knowledge, would be an alternative means of evaluating different tense logics. This approach would involve breaking down the ontological commitments of tense logics and attempting to axiomatize these logics towards an ontology of tense.

6. Conclusion

The current semantic BPM environment lacks the expressiveness for a robust ontological execution monitoring framework. In this paper, we have presented a key component in a more knowledge-focused approach to BPM. Through the creation of an ontology for compliance issue remediation, we have defined a new view of enterprise process knowledge for execution monitoring. By grounding our work in a large-scale business operation, we have provided a clear application of this framework. The identification of the status value as a key component in this framework is a significant step towards a semantics-first representation of execution monitoring and a new approach to execution monitoring. In creating representational requirements for status values, we created a specification for a new view within process ontologies. Existing work depends on semantic web technologies which lack the expressiveness to fully represent status values. The semantic robustness of the Process Specification Language was demonstrated in utilizing it as the ontological base for our representation of status values. Fulfilling these representational requirements opens up a very flexible view on processes which is important within the execution monitoring environment, but could also be extended to other applications or domains. In the future, our work will be a step towards a new kind of semantic BPM which prioritizes process semantics at all levels.

References

[1] Michael Grüninger. Ontology of the process specification language. In *Handbook on ontologies*, pages 575–592. Springer, 2004.

[2] Michael Grüninger and Mark S Fox. The role of competency questions in enterprise engineering. In *Benchmarking—Theory and practice*, pages 22–31. Springer, 1995.

[3] Martin Hepp, Frank Leymann, John Domingue, Alexander Wahler, and Dieter Fensel. Semantic business process management: A vision towards using semantic web services for business process management. In *IEEE International Conference on e-Business Engineering (ICEBE'05)*, pages 535–540. IEEE, 2005.

[4] Deliverable 1.1: Business process ontology framework. Project IST 026850 SUPER, 2007.

[5] Markus Nüttgens and Frank J Rump. syntax and semantics of event-driven process chains (epk). *Promise 2002 - Process-oriented methods and tools for the development of information systems*, 2002.

[6] Carlos Pedrinaci and John Domingue. Towards an ontology for process monitoring and mining. In *CEUR Workshop Proceedings*, volume 251, pages 76–87, 2007.

[7] Raymond Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. *Artificial intelligence and mathematical theory of computation: papers in honor of John McCarthy*, 27:359–380, 1991.

[8] Emilio M Sanfilippo, Stefano Borgo, and Claudio Masolo. Events and activities: Is there an ontology behind bpmn? In *FOIS*, pages 147–156, 2014.

[9] Antony Galin. The ontology of time and process. In *Third Interdisciplinary School on Applied Ontology*, 2016.

[10] Daya C Wimalasuriya and Dejing Dou. Ontology-based information extraction: An introduction and a survey of current approaches, 2010.

[11] Michael Grüninger. Using the psl ontology. In *Handbook on Ontologies*, pages 423–443. Springer, 2009.

[12] Boudewijn F van Dongen and Wil MP Van der Aalst. A meta model for process mining data. *EMOI-INTEROP*, 160:30, 2005.

[13] Michael Bennett. A guide to the logic of tense and aspect in english. *Logique et Analyse*, 20(80):491–517, 1977.

[14] Patrick Blackburn, Claire Gardent, Maarten De Rijke, et al. Rich ontologies for tense and aspect. *Logic, Language and Computation. CSLI, Stanford*, 1996.